Repurposing Corpora for Speech Repair Detection: Two Experiments

Simon Zwarts, Mark Johnson, Robert Dale
Center for Language Technology,
Department of Computing,
Macquarie University
{simon.zwarts, mark.johnson, robert.dale}@mq.edu.au

Abstract

Unrehearsed spoken language often contains many disfluencies. If we want to correctly interpret the content of spoken language, we need to be able to detect these disfluencies and deal with them appropriately. In the work described here, we use a statistical noisy channel model to detect disfluencies in transcripts of spoken language. Like all statistical approaches, this is naturally very data-hungry; however, corpora containing transcripts of unrehearsed spoken language with disfluencies annotated are a scarce resource, which makes training difficult.

We address this issue in the following ways: First, since written textual corpora are much more abundant than speech corpora, we see whether using a large text corpus to increase the data available to our language model component delivers an improvement. Second, given that most spoken language corpora are not annotated with disfluencies, we explore the use of Expectation Maximisation to mark the disfluencies in such corpora, so as to increase the data availability for our complete model.

In neither case do we see an improvement in our results. We discuss these results and the possible reasons for the negative outcome.

1 Introduction

We are interested in improving speech disfluency detection in transcripts of spontaneous spoken language. Many models have been proposed for this task in the literature; the best performing models so far are statistical by nature and have large data needs.

A statistical natural language processing algorithm typically has two important components: a model that describes the behaviour of interest, and the training data which is necessary to guide that model. It has been observed that simple algorithms can outperform more complex models when these simple algorithms have the advantage in terms of the amount of data available; so, for example, Brill and Banko (2001) argue that more data is more important than better algorithms for some natural language processing tasks. It is this insight that drives the work described in this paper.

Our current approach to speech disfluency detection is trained on manually-constructed spoken language corpora which contain annotations of all disfluencies as part of the transcription process. Our model is based on the noisy channel model and consists of a language model and a channel model. As we have reported elsewhere (Zwarts et al., 2010), we are able to achieve reasonable results when using Switchboard data: we obtain an F-score of 0.757 in determining which constituents of an utterance belong to a disfluency.

We would like to see if we can improve on our previously reported performance by adding more data. Our language model does not need any special annotation, and so our first set of experiments investigates whether we can improve results by vastly increasing the training data for the language model. The task of increasing the training data for the channel model is a more difficult one, since here we require the annotation of disfluencies. Our second set of experiments therefore investigates whether, given our existing annotated data, we can use Expectation Maximisation in a semi-supervised approach to automatically anno-

Simon Zwarts, Mark Johnson and Robert Dale. 2010. Repurposing Corpora for Speech Repair Detection: Two Experiments. In *Proceedings of Australasian Language Technology Association Workshop*, pages 99−106

tate a larger collection of unannotated speech data, by learning what sentences typically look like around disfluencies and what the typical structure of disfluencies is.

The remainder of this paper is structured as follows. In Section 2 we first present some background on disfluencies and their structure in spontaneous speech. Section 3 discusses the current state of the art in disfluency detection models, motivates the choice of the model we use, and describes some of its intricacies and the data sets we use. Section 4 investigates the language model component of our model and explores whether we can improve this component; we provide the results obtained when using a language model that is several orders of magnitude larger than the language model used in our previous work. Section 5 investigates a more radical approach to address our data needs: we alter the training data for both the language model and the channel model.

It turns out that neither of these experiments results in an improvement in disfluency detection. Section 6 draws some conclusions from our results, and suggests some ways forward based on this experience.

2 Speech Repairs

We adopt the terminology and definitions introduced by Shriberg (1994) to discuss disfluencies. We are particularly interested in those disfluencies which are categorised as **repairs**. These are the most interesting and also the hardest disfluencies to identify, since they are not marked by a characteristic vocabulary. Shriberg (1994) identifies and defines three distinct parts of a such a disfluency, referred to as the **reparandum**, the **interregnum** and the **repair**. Consider the following utterance:

$$\begin{gathered}\textit{I want a flight } \overbrace{\textit{to Boston,}}^{\text{reparandum}} \\ \underbrace{\textit{uh, I mean}}_{\text{interregnum}}\underbrace{\textit{to Denver}}_{\text{repair}} \textit{ on Friday}\end{gathered} \tag{1}$$

The reparandum *to Boston* is the part of the utterance that is being 'edited out'; the interregnum *uh, I mean* is a filler, which may not always be present; and the repair *to Denver* replaces the reparandum.

Given an utterance that contains such a disfluency, we want to be able to correctly detect the start and end positions of each of these three components. We can think of each word in an utterance as belonging to one of four categories: fluent material, reparandum, filler, or repair. We can then assess the accuracy of techniques that attempt to detect disfluencies by computing precision and recall values for the assignment of the correct categories to each of the words in the utterance, as compared to the gold standard as indicated by annotations in the corpus.

3 Disfluency Detection Models

3.1 Related Work

A number of different techniques have been proposed for automatic disfluency detection. Schuler et al. (2010) propose a Hierarchical Hidden Markov Model approach; this is a statistical approach which builds up a syntactic analysis of the sentence and marks those subtrees which it considers to be made up of disfluent material. Although this is one of the few models that actually builds up a syntactic analysis of the utterance being analysed, its final F-score for fluency detection is lower than that of other models.

Snover et al. (2004) investigate the use of purely lexical features combined with part-of-speech tags to detect disfluencies. This approach is compared against approaches which use primarily prosodic cues, and appears to perform equally well. However, the authors note that this model finds it difficult to identify disfluencies which by themselves are very fluent. The edit repairs which are the focus of our work typically have this characteristic: when a speaker edits her speech for meaning-related reasons, rather than errors that arise from performance, the resulting disfluency can be by itself fluent. We can see this in Example (1): the repair and the reparandum are equally fluent. This makes it difficult to distinguish reparanda as being part of disfluencies when only lexical cues are available. Since the transcripts we work with do not have prosodic cues annotated, we need to look elsewhere for a solution to this problem.

Noisy Channel models have done very well in this area; the work of Johnson and Charniak (2004) explores such an approach. This approach performs very well when compared with other approaches. Johnson et al. (2004) adds some handwritten rules to the noisy channel model, providing the current state of the art in disfluency detection. Lease and Johnson (2006) also use this approach, but they are particularly interested in finding fillers; they use early filler detection and deletion in this model.

The following section describes the noisy channel approach in more detail.

3.2 The Noisy Channel Approach

The approach we build on is that first introduced by Johnson and Charniak (Johnson and Charniak, 2004). This approach is modular by nature, making it possible to interchange different sub-components. The original paper explores the use of different types of language models, and demonstrates how some models provide better overall performance than others. In the remainder of this section we describe the basics of this approach.

To find repair disfluencies, a noisy channel model is used. For an observed utterance with disfluencies y, we wish to find the most likely source utterance, $\hat{x}$, where:

$$\begin{aligned} \hat{x} &= argmax_x\ p(x \mid y) \\ &= argmax_x\ p(y \mid x)\, p(x) \end{aligned} \tag{2}$$

Here we have a channel model $p(y|x)$ which generates an utterance y given a source x and a language model $p(x)$. We assume that x is a substring of y, i.e., the source utterance can be obtained by marking words in y as being disfluent elements and effectively removing them from this utterance.

The task of the language model is to assess the fluency of the sentence when the reparandum and the interregnum have been removed. As noted above, Johnson and Charniak (2004) experiment with variations on the language model; they report results for a bigram model, a trigram model, and a language model using the Charniak Parser (Charniak, 2001). Their results demonstrate that the parser model outperforms the bigram model by 5%.

The channel model is based on the intuition that a reparandum and a repair are generally very alike; it is often the case that the repair is almost a copy of the reparandum. In the training data, over 60% of the words in a reparandum are lexically identical to the words in the corresponding repair. Example (1) again provides an example of this: half of the repair is lexically identical to the reparandum. The channel model therefore gives the highest probability when the reparandum and repair are identical. When the potential reparandum and potential repair are not identical, the channel model performs deletion, insertion or substitution operations. The probabilities for these operations are defined on a lexical level and are derived from the training set text. This channel model is formalised using a Synchronous Tree Adjoining Grammar (S-TAG) (Shieber and Schabes, 1990), which matches words from the reparandum to the repair. The weights for these S-TAG rules are learnt from the training text, where reparanda and repairs are aligned to each other using a minimum edit-distance string aligner.

For a given utterance, every possible utterance position might be the start of a reparandum, and every given utterance position thereafter might be the start of a repair (to limit complexity, a maximum distance between these two points is imposed). Every disfluency in turn can have an arbitrary length (again up to some maximum to limit complexity). After every possible disfluency other new reparanda and repairs might occur; the model does not attempt to generate crossing or nested disfluencies, although they do very occasionally occur in practice. To find the optimal selection for reparanda and repairs, all possibilities are calculated and the one with the highest probability is selected.

A chart is filled with all the possible start and end positions of reparanda, interregna and repairs; each entry consists of a tuple $\langle rm_{\text{begin}}, ir_{\text{begin}}, rr_{\text{begin}}, rr_{\text{end}} \rangle$, where rm is the reparandum, ir is the interregnum and rr is the repair. A Viterbi algorithm is used to find the optimal path through the utterance, ranking each chart entry using the language model and channel model. The language model, a bigram model, can be easily calculated given the start and end positions of all disfluency components. The channel model is slightly more complicated because an optimal alignment between reparandum and repair needs to be calculated. This is done by extending each partial analysis by adding a word to the reparandum, the repair or both. The start position and end position of the reparandum and repair are given for this particular entry. The task of the channel model is to calculate the highest probable alignment between reparandum and repair. This is done by initialising with an empty reparandum and repair and 'growing' the analysis one word at a time. Using a similar approach to that used in calculating the edit-distance between reparandum and repair, the reparandum and repair can both be extended with one of four operations: deletion (only the reparandum grows), insertion (only the repair grows), substitution (both grow), or copy (both grow). When the reparandum and the repair have their length corresponding to the current entry in the chart, the channel probability can be calculated. Since there are multiple alignment possibilities, we use dynamic programming to select the most probable solutions. The probabilities for insertion, deletion and substitution are estimated from the training corpus. We use a beam-search strategy to find the final optimum when combining the channel model and the language model.

3.3 The Data Set

As a data set to work with, we use the Switchboard part of the Penn Treebank 3 corpus. The Switchboard Corpus is made up of transcriptions of spontaneous conversations between two partners during a telephone call. The Penn Treebank 3 corpus adds manual annotation of disfluencies to the Switchboard corpus; additionally it provides part-of-speech information for all the words.

The disfluency annotation distinguishes between repair disfluencies and filled pauses. When repair disfluencies are present the structure of the disfluency is annotated: these annotations indicate which part of the disfluency is the reparandum, which part is the interregnum and which part is the repair. The following is an example:

```
  [ i/NN think/VBP it/PRP was/VBD +
{F yeah/UH } i/NN think/VBP that/WDT
was/VBD ] the/DT only/JJ question/NN
E_S
```

Here we see the reparandum (*I think it was*), the interregnum (*yeah*) and the repair (*I think that was*) annotated.

Following Johnson and Charniak (2004), we use all of sections 2 and 3 of the corpus for training; we use conversations 4[5-9]* for a held-out training set; and conversations 40*, 41[0-4]* and 415[0-3]* as the held-out test set.

The corpus is not immense: a little over 100K sentences are present in the training data. This means that in the the held-out training set, and presumably also in the test set, there are many out-of-vocabulary words and a very large incidence of low frequency vocabulary items, for which we struggle to find the appropriate statistical values.

Our earlier work just used this data. When we use the noisy channel model as described in Section 3.2 using the Switchboard data, as described above, we can compute precision and recall over a held-out test set. Comparing our output against the gold standard annotation, we can compute performance over disfluencies detected. This results in an F-score of 0.757.[1]

4 Extending The Language Model

4.1 Background

As we noted earlier, previous work by Johnson and Charniak (2004) has shown that the language model component of the model has an important role: when more sophisticated language models are used, the overall performance can be increased significantly.

An important aspect of our earlier work is that we were particularly interested in processing incoming speech incrementally, detecting disfluencies as soon after they happen as possible. However, incremental processing makes the use of a reranker, as adopted in Johnson and Charniak's more sophisticated model, a less viable option. Our initial language model was trained on the fluent part of the Switchboard Corpus: this consists of the utterances with the reparanda and the interregna removed. The bigram model is trained on the counts from the same data and, as mentioned above, this contains approximately 100k sentences. This would not typically be considered a large data set in terms of language modelling

[1] The F-score reported here is the harmonic mean between precision and recall.

(Harb et al., 2009); consequently, we look to increasing the amount of data used in our language model as an alternative means of improving results.

4.2 Motivation

Using a larger set of data for the language model allows us to answer two questions:

1. Has the current bigram model reached its limit? Previous research has shown that reranking the results of a model using bigrams still leaves room for improvement. We assume that the bigram model itself also has scope for improvement, since there is still a large set of out-of-vocabulary words in the held-out training set, and an even larger set of low frequency words for which it is difficult to calculate the proper probabilities accurately. Can we improve the bigram model when we increase its training data?

2. Does the nature of the data used matter? Our language model is currently specifically trained on the fluent parts of transcribed spontaneous speech. Most language models, however, are built on primarily written texts, given their greater availability. Would the use of a vastly greater quantity of written data offset the impact of the change in the nature of that data?

4.3 Experimental Setup

We decided to use the Google Web 1T corpus, which contains English word n-grams and their observed frequency counts. The n-gram counts were generated from approximately 1 trillion word tokens of text from publicly accessible Web pages, much larger than the number of words in Switchboard (roughly 700K). In their description of this corpus, the authors suggest the corpus should be useful for language models and for speech recognition; our experiments are one test of this claim.

The Web 1T corpus records counts for unigrams up to 5-grams. We only use the bigram part of this corpus, but this still introduces memory problems. The entire bigram counts take up more than 8.8GB, which is more than we can fit into memory. This dataset is also vastly larger than the test portion of the corpus. Since our evaluation is only carried out over the test portion, we do not need to memorise any bigrams which are not present in this portion; so, we can use the process of prefiltering (Goodman, 2001) the bigrams of the larger corpus against the test set. This process does not mean we are using test data during our experiments: it is only an optimisation strategy that avoids loading into memory bigrams which will not be used later. After this process of prefiltering we are left with only 10MB of bigram data, which easily fits into memory.

Our baseline model is the model as described by Johnson and Charniak (2004), using the traditional Switchboard part of the Penn Treebank 3 data to derive the language model. Our alternative model has the language model replaced with the Web 1T bigram probabilities. If this approach proves to be successful, we might consider using a language model which is a hybrid consisting of both the data derived from the Switchboard part (which is arguably closer in nature to the data we ultimately want to process), and the Web 1T data (which might deliver statistics for the tail end of the Zipfian curve). We can use the held-out training set for tuning purposes to decide on the relative weight to be accorded to these two language models.

4.4 Results

The baseline model, using only the Switchboard data with a bigram language model, results in an F-score of 0.757. Our new model, which uses a vastly larger data set for bigram modelling, results in an F-score of 0.739.

The most obvious explanation for this is that text derived from Web pages is not a good source of data for building a language model for spoken language: Even when disfluencies are removed from spontaneous spoken speech, the language used is still very different from written text. In general terms, this, of course, is not a new or surprising result; Biber (1988), and many others since, have drawn attention to the differences between spoken and written language. What is perhaps more surprising is that these differences appear to impact not only, for example, at the syntactic level, but also at the level of bigram occurrences.

5 A Semi-supervised Learning Approach

5.1 Background

Our noisy channel approach has two components, the language model and the channel model. The approach in the previous section investigate whether it would be possible to use a very large data set for the language model. In this section we investigate whether it is possible to address the data needs for both the language model and the channel model.

5.2 Motivation

Our objective here is to use a data set of transcribed spontaneous speech which is more than an order of magnitude larger than the data available in the Switchboard part of the Penn Treebank 3 corpus. With this approach we would hope to answer the following three questions:

1. Is it possible to significantly increase the performance of this model, without the application of a more complicated approach? As noted above, complications like reranking via parser results are difficult to apply in our incremental processing scenario.

2. What does the performance curve of this model look like? When we increase training data, how does the overall performance increase? Our interest here is in providing a more definitive assessment as to how much data is needed to reach the upper limit of performance with the current model.

3. Can we use a Expectation Maximisation approach in order to increase our data needs? Disfluency-annotated data is very costly to develop; we want to see whether we can avoid this by automatically deriving such annotations using a semi-supervised approach.

5.3 Experimental Setup

In the experiments described here, we explore increasing the training data by using additional speech corpora.

The Fisher English Training Speech Transcripts represent the collection of conversational telephone speech (CTS) that was created at the LDC during 2003. It contains transcript data for 5,850 complete conversations, each lasting up to 10 minutes. The Fisher Speech Corpora Part I and II together contain a little over 2 million sentences, which is considerably more than is present in the Switchboard part of Penn Treebank 3. However, the only disfluency annotation the corpus contains is the marking of partially uttered words. Filled pauses and the more complicated repair disfluencies are not annotated.

Besides lacking disfluency annotations, the Fisher corpora also lacks part-of-speech tags. Our channel model uses these tags to build up an alignment between reparandum and repair: since it assumes reparandum and repair are a rough copy of each other, it uses the part-of-speech tags to inspect how similar these parts are, and these tags are especially useful when the words in the reparandum and repair are not exact lexical copies. Since it is too costly to obtain manually-annotated tags for our corpus, we use the Brill Tagger (Brill, 1993) to automatically annotate the Fisher corpus with part-of-speech tags, using the same tag set as is used in the Penn Treebank 3 data.

Once the part-of-speech tags are available, we can use our original noisy channel model to annotate this corpus for disfluencies. We can then add this newly acquired data to the existing training data. In this way, we hope to acquire new statistical insights into what types of disfluencies are common, and what sentences typically look like around these disfluencies.

In order not to dominate the manually-annotated data from the Penn Treebank 3 data with the more noisy Fisher data, we would as a first step like to use them in similar proportions. We initially only use the first part of the Fisher data, of a similar size to the Penn Treebank 3 data. When this approach results in increased performance, we can re-annotate this same part with the newly built model, which hopefully will result in a better analysis of the Fisher corpus. When this iterative process reaches its maximum score, we can then investigate whether we can use more of the Fisher data. Because the original Penn Treebank 3 data is hand-annotated and is more accurate, it might prove to be helpful to not weight counts from both corpora equally: doing so might make the model drift away from disfluency detection to another annotation scheme which fits the data better, but which ultimately could be meaningless. We can use the held-out training data to properly decide on a weighting scheme between both corpora.

The baseline which we compare against is the standard model as described by (Johnson and Charniak, 2004), using the Penn Treebank 3 data set only.

5.4 Results

The baseline model using only the Switchboard data part results in an F-score of 0.757 using the bigram language model. When we add the Fisher data as part of our training data we expect to achieve a higher performance; however in our experimental set-up we reached a final F-score of 0.742, which is actually a slight decrease in performance. This is disappointing, since Expectation Maximisation has proven to be a successful strategy in other area of natural language processing.

There are several possible reasons as to why this approach turned out to be less fruitful here. First, note that the training process heavily relies on part-of-speech information. However, the Brill Tagger was not initially built for spontaneous speech, and may have introduced errors which impact on our final results. An alternative explanation could be that the Fisher corpus and Switchboard corpus exhibit a different type of language use, although this seems to be less likely. Finally, it could be the case that our model does not perform well enough on the Fisher data to actually help out in a new iteration, although for the expectation maximisation step an F-score of around 0.75 should not be a hindrance to building a new model for a next iteration. Significant gains using Expectation Maximisation have been achieved in other spoken language processing tasks starting from this absolute score (Sandrini and Federico, 2003). We are not yet convinced, therefore, that this direction is a dead-end.

6 Conclusions and Future Work

Statistical models are typically data-hungry, and so a problem arises in any domain where data is scarce. In this paper, we have explored two different approaches that aim to increase the amount of data usable by our disfluency detection model. We have investigated the use of Google 1T, the largest written text corpus available to date for language modelling. This proved to have a negative impact on our results. We hypothesise that this is most likely because of the differences between written and spoken language. The result means that one should be cautious about using corpora derived from textual sources when working with conversational speech.

In our second set of experiments, we tried to use Expectation Maximisation to provide more data for use in our channel model. Again, the results here were negative.

Ultimately, although it may be true that more data can be more important than smarter algorithms, it needs to be the right data.

For future work we intend to experiment with a different part-of-speech tagger. We also suspect that a different source of data may require retuning of our model: currently our model is trained towards the Switchboard data, and even though this is the only data for which we have gold standard annotations, we would like to retune the model parameters when using the Fisher corpus. We can still use the held-out Switchboard data set to retune the model operating on Switchboard and Fisher. The current approach uses a noisy channel model, in which the language model and channel model are weighted equally. We could transform this into a log linear model which will allow us not only to weight the language model and channel model differently, but also will allow us to use multiple models. We can develop separate language models from different sources (Web1T, Fisher, Switchboard) and separate channel models derived from different sources (Fisher via EM training, Switchboard) and use them simultaneously. Using a log linear approach we can individually weight these components using the held-out training set to achieve optimal performance. This almost guarantees that performance will not degrade, as in a worst case scenario the learner can turn off new data sources and use the old model; but even when there is a little information in any of the additional sources, performances is expected to go up. Finally, using such a model will allow us to add any computable feature, making it possible to go beyond language and channel models. As an additional advantage, the individual learnt weights will be a good indication of the relative value of each data source.

Acknowledgements

This work was supported by the Australian Research Council as part of the Thinking Head Project, ARC/NHMRC Special Research Initiative Grant # TS0669874. We thank the anonymous reviewers for their helpful comments.

References

Douglas Biber. 1988. *Variation across speech and writing*. Cambridge University Press.

Erik Brill and Michele Banko. 2001. Mitigating the Paucity-of-Data Problem: Exploring the Effect of Training Corpus Size on Classifier Performance for Natural Language Processing. In *Proceedings of the First International Conference on Human Language Technology Research*.

Eric Brill. 1993. *A corpus-based approach to language learning*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA, USA.

Eugene Charniak. 2001. Immediate-head parsing for language models. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 124–131.

Joshua T. Goodman. 2001. A bit of progress in language modeling. Technical report, Microsoft Research.

Boulos Harb, Ciprian Chelba, Jeffrey Dean, and Sanjay Ghemawat. 2009. Back-Off Language Model Compression. In *Proceedings of Interspeech*, pages 325–355.

Mark Johnson and Eugene Charniak. 2004. A tag-based noisy channel model of speech repairs. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 33–39.

M. Johnson, E. Charniak, and M. Lease. 2004. An improved model for recognizing disfluencies. In *in Proceedings of Conversational Speech Rich Transcription Fall Workshop*.

Matthew Lease and Mark Johnson. 2006. Early deletion of fillers in processing conversational speech. In *Proceedings of the Human Language Technology Conference of the NAACL*, pages 73–76.

Vanessa Sandrini and Marcello Federico. 2003. Spoken Information Extraction from Italian Broadcast News. *Advances in Information Retrieval Lecture Notes in Computer Science*, 2633.

William Schuler, Samir AbdelRahman, Tim Miller, and Lane Schwartz. 2010. Broad-Coverage Parsing using Human-Like Memory Constraints. *Computational Linguistics*, 36(1):1–30.

Stuart M. Shieber and Yves Schabes. 1990. Synchronous tree-adjoining grammars. In *Proceedings of the 13th International Conference on Computational Linguistics*, pages 253–258.

Elizabeth Shriberg. 1994. *Preliminaries to a Theory of Speech Disuencies*. Ph.D. thesis, University of California, Berkeley.

Matthew Snover, Bonnie Dorr, and Richard Schwartz. 2004. A Lexically-Driven Algorithm for Disfluency Detection. In *Proceedings of Human Language Technologies and North American Association for Computational Linguistics*, pages 157–160.

Simon Zwarts, Mark Johnson, and Robert Dale. 2010. Detecting speech repairs incrementally using a noisy channel approach. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1371–1378, Beijing, China, August. Coling 2010 Organizing Committee.